SRI International

PARALLEL GUESSING: A STRATEGY FOR HIGH-SPEED COMPUTATION

Technical Note No. 338

September 19, 1984

By: Martin A. Fischler, Program Director, Perception
Oscar Firschein, Staff Scientist

Artificial Intelligence Center
Computer Science and Technology Division

Presented to the Workshop on Algorithm-Guided Parallel Architectures for Automatic Target Recognition [Sponsored by the DARPA Tactical Technology Office (TTO), the Naval Research Laboratory, and the Army Night Vision and Electro-Optics Laboratory (NVL)] July 16-18, 1984 in Leesburg, VA.

SRI Project 5355
Contract MDA903-83-C-0027

SRI International®

333 Ravenswood Ave. • Menlo Park, CA 94025
(415) 326-6200 • TWX: 910-373-2046 • Telex: 334-486

CONTENTS

ABSTRACT

Attempts have been made to speed up image-understanding computation involving conventional serial algorithms by decomposing these algorithms into portions that can be computed in parallel. Because many classes of algorithms do not readily decompose, one seeks some other basis for parallelism (i.e., for using additional hardware to obtain higher processing speed). In this paper we argue that "parallel guessing" for image analysis is a useful approach, and that several recent IU algorithms are based on this concept. Problems suitable for this approach have the characteristic that either "distance" from a true solution, or the correctness of a guess, can be readily checked. We review image-analysis algorithms having a parallel guessing or randomness flavor.

We envision a parallel set of computers, each of which carries out a computation on a data set using some random or guessing process, and communicates the "goodness" of its result to its co-workers through a "blackboard" mechanism.

I INTRODUCTION

Sophisticated image analysis often requires the use of a sequence of time-consuming algorithms, such as feature extraction, region growing, and model instantiation. Such processing sequences currently require several minutes for their computations on commonly available machines, see Table 1. "Real-time" scene analysis with frame rates of 1/30 second will require three or four orders of magnitude speedup. If future computer technology advances provide us with one or two orders of magnitude over the next 5 to 10 years, two or three orders of magnitude improvement are still required for practical applications in the indicated 5-10 year period.

Table 1
Timing for Some Image Understanding Algorithms

(all timing is CPU time of a VAX 11/780)	
RELAX relaxation algorithm, University of Maryland, (3x3 window, 128x128 image)	3 minutes/iteration
Phoenix segmentation algorithm, Carnegie Mellon university, 500x500 image	33 minutes
GHOUGH, generalized Hough Transform, University of Rochester (variable, depending on image size, number of rotations and radii tried, and template size)	1-5 minutes

In recent years, parallel architectures for image processing have been developed; a recent survey of these is given in [Reeves 1984] and in [Duff 1983]. These architectures are largely tailored for the natural parallelism found in convolution, filtering, and other "low-level" scene analysis processes. However, higher-level processes do not exhibit such parallelism and, in general, algorithmic parallelism cannot be achieved by attempting to decompose essentially sequential algorithms. (Shannon showed this for the case of n-dimensional switching functions, [Shannon 1949]).

We therefore seek a generally applicable formalism for image analysis algorithms that offers a natural parallelism, so that we can trade additional hardware for decreased computation time. One class of such algorithms takes advantage of the following observation: *It is often much faster to verify the*

correctness of a guess, than to compute the solution. Based on this observation, we postulate an architecture based on a large set of processors that guess an answer (1) by means of random selection, (2) by an exhaustive "rough grain" selection, or (3) by *intelligent* guessing. Such guessing mechanisms become especially important in problems in which the data are noisy, or when there is not an adequate analytical model.

II APPROACH TO PARALLELISM

Our approach is, therefore, to develop image analysis algorithms suitable for parallel computation that are based on guessing a good answer. The basic idea is that each module simultaneously takes a different guess and computes a "goodness" value for the guess. When a "good" guess is made, its result and the goodness value are entered on a "blackboard." The blackboard controller indicates the basis for further iterations by constraining the range of new values to be chosen and determines when a suitable answer has been found.

In contrast to most current concepts, based on a small "grain size" and high-bandwidth communication between processing modules, we seek algorithms that do not require lockstep operation of processors and require a minimum of communication between processors. The proposed parallel architecture is shown in Figure 1. Symbolic structures derived from low-level processing are stored in a *blackboard.* Parallel processors derive their input data from this blackboard directed by a control processor. Intermediate results are returned by the processors to the blackboard. Results of the computation are analyzed by the control processor and then used as output. The requirements for algorithms to be used in this architecture are as follows:

- **Selection.** A selection method for the guess must be provided, using intelligent guessing, a random selection, or exhaustive selection from a roughly quantized space. The selection process can be carried out in *data space* by selecting from among the input data, in *parameter* space in which there is a selection of values for one or more of the model parameters, or in both data and parameter space.
- **Goodness of result.** There must be a simple measure of the goodness of the result obtained using the guess.
- **Control.** Some method must be provided for selecting the best current guesses, for using the best current guess to constrain additional guessing, for accomplishing efficient guessing by partitioning the space of guesses, and for determining when the overall process is to stop.

III EXAMPLE ALGORITHMS

Many existing scene-analysis-related algorithms can be viewed as satisfying the above requirements. The Hough transform, RANSAC, back-projection techniques, branch-and-bound, and functional optimization are particular examples. These are described below.

A. Hough Transform Approach

The Hough transform [Duda and Hart, 1972] can be used when little is known about the scale and location of the boundary of an object we wish to find, but its shape can be described by a parametric curve, e.g., a straight line. This class of algorithms finds the parameters of a model within a roughly quantized range of variable values in the equations of the model. For example, if we are given a list of edge pixels and wish to find an acceptable straight line that passes through or near many of these pixels, we can use the approach shown in Figure 2.

This approach is mechanized using random data selection. Each processor selects an edge point from the list and considers a span of lines of various directions through the point. The distance of the normal to each such line is computed, and the value of the normal distance and angle for each line is used to increment an appropriate *(angle, normal distance)* histogram "bucket" on the blackboard. A control processor associated with the blackboard stops the process when it determines that a histogram peak is evident and returns the *(angle, normal distance)* value of the peak as the parameters of the desired line.

1. *RANSAC*

The random selection and consensus (RANSAC) approach [Fischler and Bolles, 1981] is a procedure that uses a random selection of exactly enough data points to satisfy a model. Each trial involves random point selection and testing of the proposed model on the remaining points. A simple example of RANSAC is the case of determining an acceptable line, given a set of candidate edge points. A pair of points is randomly selected, as shown in Figure 3, and the sum of the absolute values of the deviations of the other points from this line is used as a measure of goodness of fit.

The algorithm uses random selection of data points. In the parallel mechanization, each processor selects a pair of input points, computes the line parameters, and then determines the sum of the deviations of the other points from this line. Each processor looks at the blackboard to see if the sum of the deviations obtained is less than the best value posted on the blackboard. If it is, then the previous value is replaced by the new value and the line parameters. When the sum of the deviations is less than a desired amount (and the individual deviations are not correlated in any way), the control processor stops the process by writing a termination message on the blackboard.

2. *Back Projection*

In the "back-projection" problem, we are given an image and want to determine the structure of the scene that produced the image. Witken [Witken 1981] finds the 3-D orientation of small planar patches within the scene to obtain an estimate of the 3-D geometry of objects in the scene. His approach makes the following assumption: *An arbitrary scene will have no favored direction for its visible edges.* The algorithm first finds the edges in the image and then finds the tangent lines to these edges. A trial and error procedure of assuming specific planar patch orientations in the scene is now carried out. The "best" planar patch for each local region in the scene is the one for which the distribution of the "back-projected" line orientations will be "most random."

In the parallel mechanization shown in Figure 4, each processor obtains, from the blackboard, a list of tangent lines in some local patch of the image. Each processor accepts a complete set of scene planar patch orientation parameters specified on the blackboard and uses these parameters to back-project the tangent lines. Each processor develops a histogram of line directions, for each trial orientation of the scene patch, as an indicator of the randomness for the line directions: the flatter the histogram, the better the estimate. Each processor reports to the blackboard the best orientation of the patch it is analyzing and the goodness of the result. The blackboard control computer specifies the range of plane orientations to use, assigns portions of the image to the computers for analysis, and decides when to stop the process.

3. *Branch-and-Bound*

Branch-and-bound is a popular technique that has been used successfully in the solution of problems that arise in combinatorial optimization and artificial intelligence. In a branch-and-bound approach, the solution space is organized as a graph that is usually a tree, with each link representing a value. The goal is to proceed from the root node to some end node in a way that maximizes or minimizes the sum of the path values. Various forms of branch-and-bound are all based on the idea of avoiding paths that are unproductive. Thus, one might begin by following a random path from the root to the goal to obtain a total cost C for that path. One then explores another path, stopping the exploration of that path when the path cost exceeds C. If a lower-total-cost path from root to goal is found, then its cost becomes the new C. Parallelism can be introduced into the process by expanding more than one path during each iteration. A parallel computer for implementing branch-and-bound algorithms is given in [Wah and Ma 1982], and the problems that arise in parallel branch-and-bound are given in [Lai and Sahni 1984].

Using the multiple-path-expansion approach, each processor follows a path, computing the cumulative cost as it proceeds. When a processor has followed a path from root to goal, it posts the total cost on the blackboard. Any processor that currently has a greater cost terminates its current path and pursues a new path. This procedure is shown in Figure 5.

4. *Maximum or Minimum of a Function*

There are many image-analysis applications involving the determination of the maximum or minimum of a function. If the function is relatively smooth, then an iterative gradient approach can be used in which a measure of the gradient is used to determine the next guess as to the independent variable. If the function has many local maxima or discontinuities, a *coarse-fine* approach is more appropriate, in which random exploration is carried out in coarse partitions of the independent variable, and a finer exploration is then made in locations that seem promising.

In the *coarse-fine* parallel mechanization, each processor is assigned a range and makes random guesses of the independent variable within its assigned range. After n random looks, only the most promising ranges are retained, and the processors are redistributed to cover the selected ranges. The random guessing procedure is continued for a number of iterations in the selected ranges, and sub-ranges are identified for further exploration. The motivation is to avoid getting trapped in a local maximum at an early stage of the process. In the parallel mechanization shown in Figure 6, we use a simple one-dimensional example of finding a global maximum, given a "noisy" function having many local maxima.

IV CONCLUSIONS

Guessing techniques based on randomness or exhaustive bucketing can be important in image analysis, since such guessing is often needed in the face of data errors or lack of a suitable analytic model. In addition, guessing offers the advantage of a uniform approach to achieving parallelism. We have indicated a parallel architecture that can take advantage of such an approach and some present-day algorithms that can be viewed from this point of view. Rethinking of some of the "classical" image-analysis algorithms in this context can prove fruitful.

REFERENCES

Duda, R.O. and P.E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," Comm ACM, 15,1, Jan. 1972, 11-15.

Fischler, M.A. and R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," Comm. ACM Vol. 24(6), pp. 381-395 (June 1981).

Frisby, J.P., "Seeing: Illusion, Brain, and Mind," Oxford University Press, 1980.

Lai, T. and S. Sahni, "Anomalies in Parallel Branch-and-Bound Algorithms," Comm. of the ACM, June 1984, Vol. 27, No. 6.

Reeves, A. P., "Parallel Computer Architectures for Image Processing," Computer Vision, Graphics, and Image Processing 25, 68-88(1984).

Shannon, C.E., "Synthesis of two-terminal switching networks," Bell System Technical Journal, Jan. 1949 28(1):59-98.

Wah, B. and Ma, Y., "NANIP - A parallel computer system for implementing branch-and-bound algorithms." Proc. 8th Ann. Symp. on Computer Architecture, 1982, pp. 239-262.

Witkin, A. P., "Recovering surface shape and orientation from texture," Artificial Intelligence 17, 1981, 17-47.

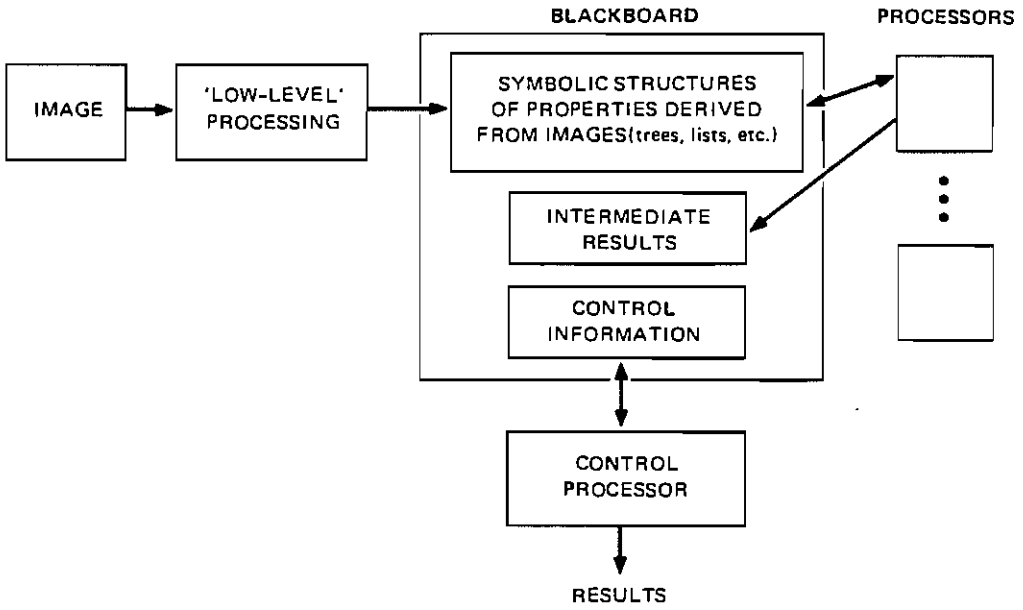

FIGURE 1 PARALLEL PROCESSING ARCHITECTURE FOR GUESSING ALGORITHMS

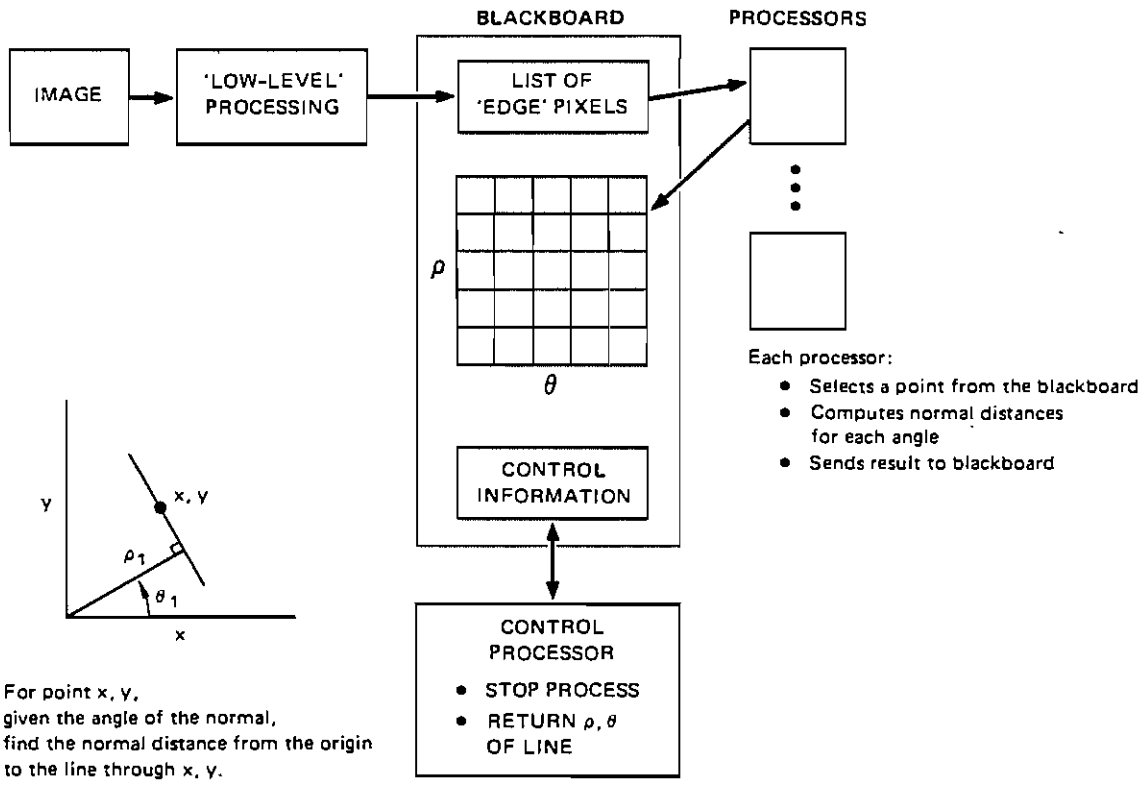

FIGURE 2 HOUGH APPROACH

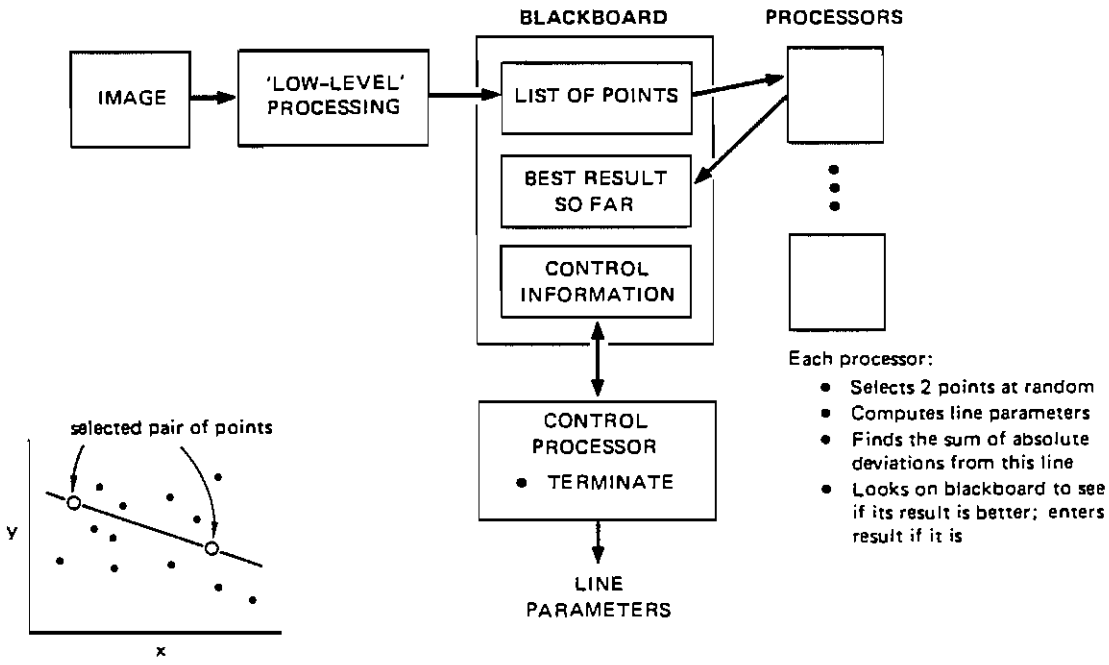

FIGURE 3 RANSAC:RANDOM SELECTION AND CONSENSUS

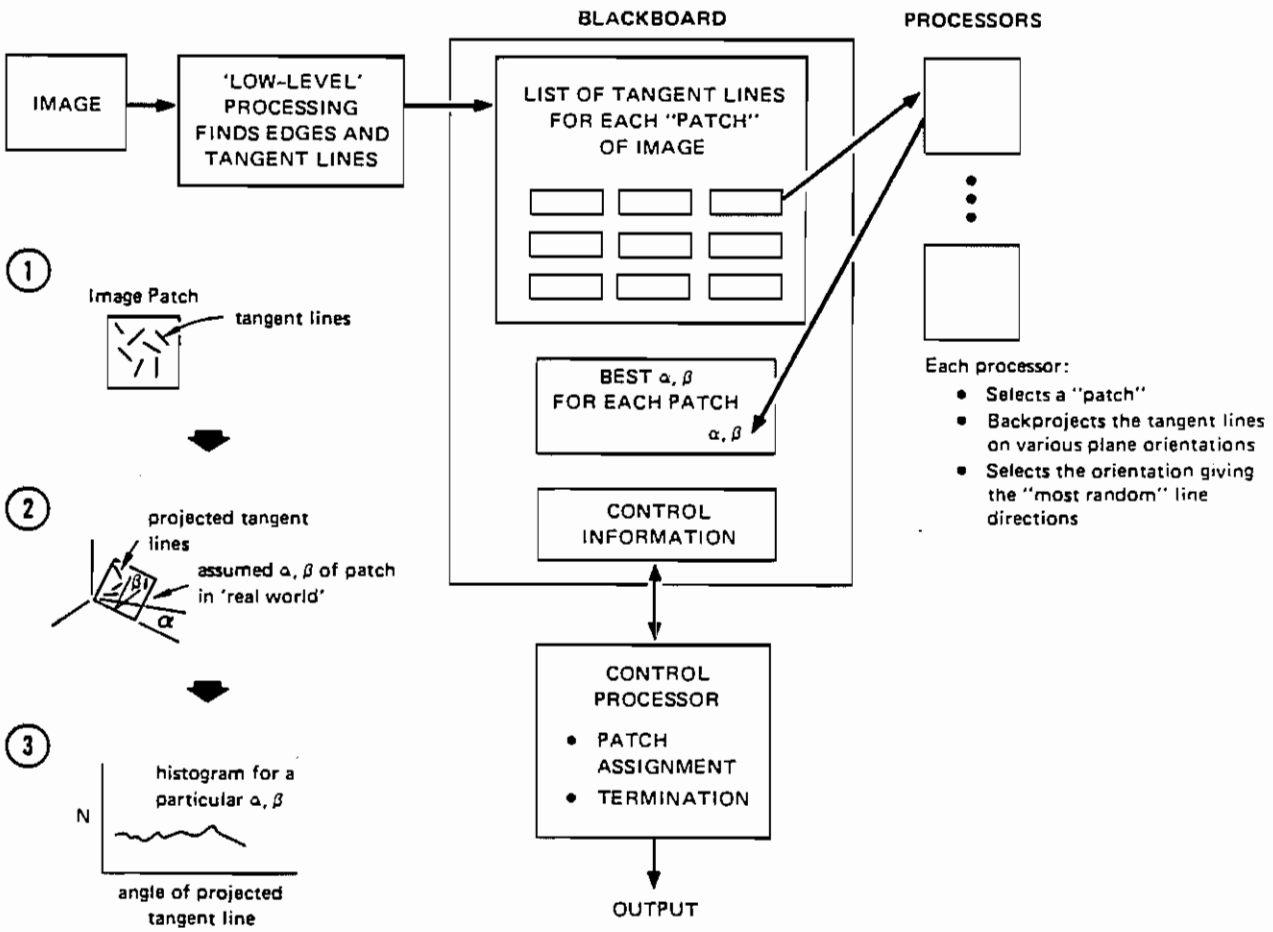

FIGURE 4 BACK-PROJECTION

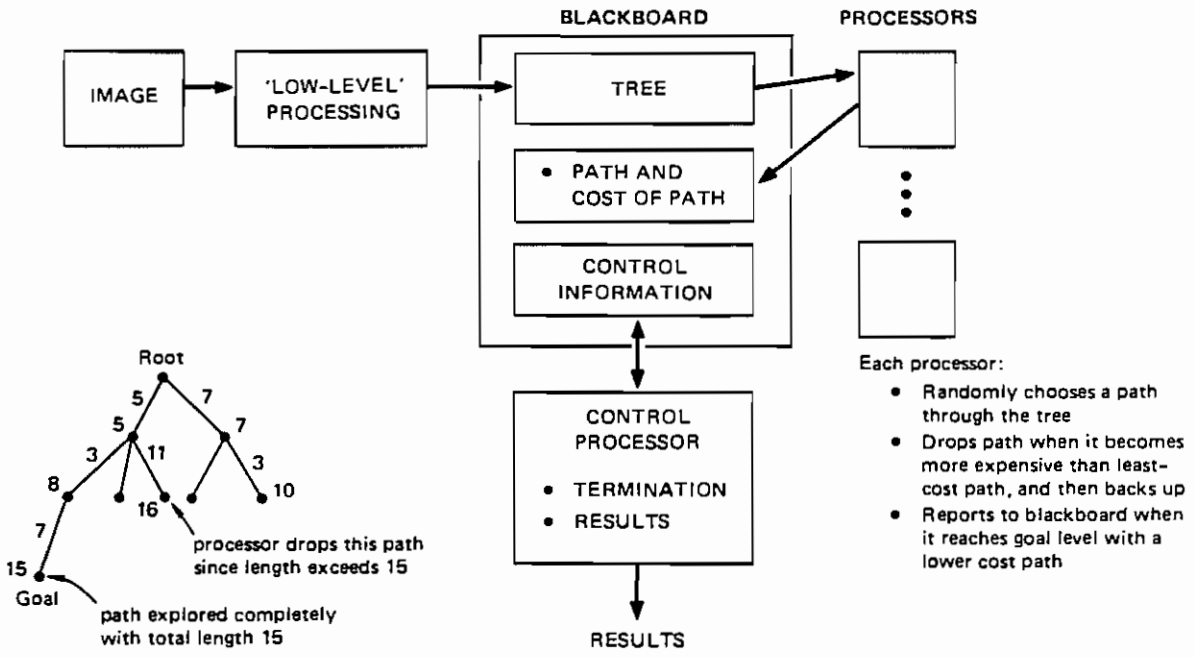

FIGURE 5 BRANCH AND BOUND

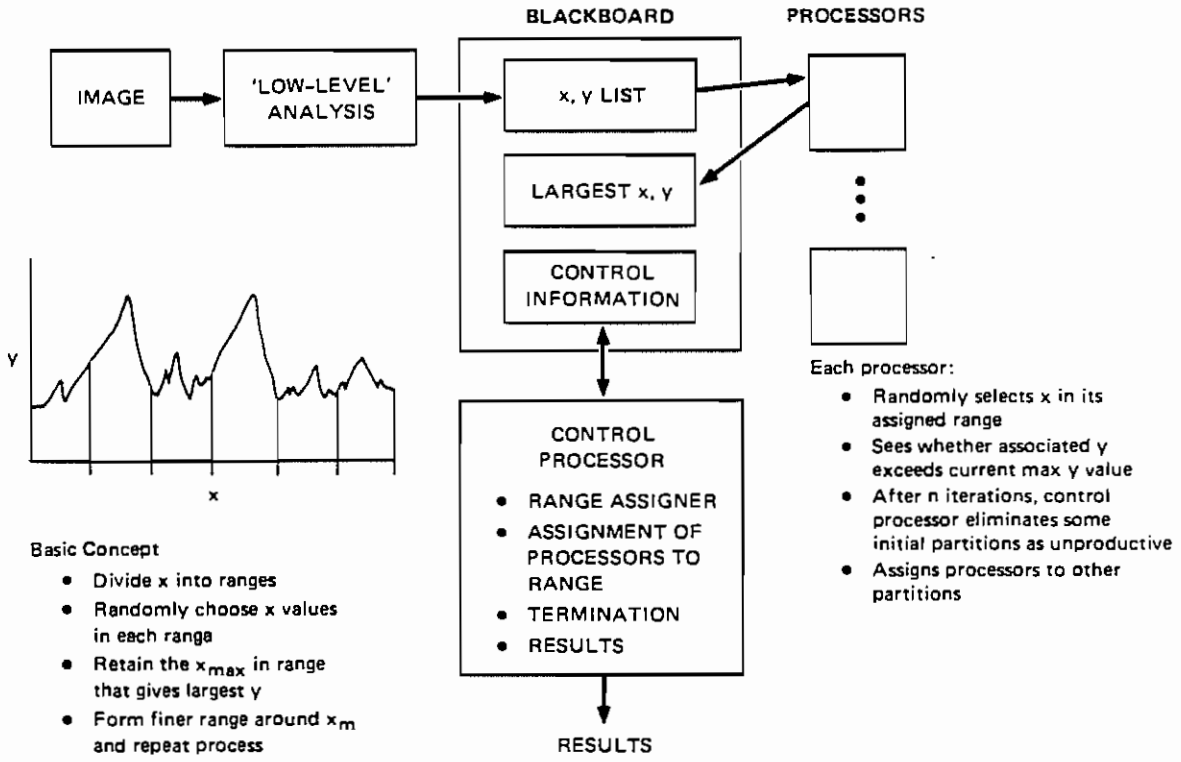

FIGURE 6 COARSE-FINE, MAX-MIN OF A FUNCTION